

# SysLink 02.00.00.68 beta1 ReleaseNotes

---

## SYS/Link 02.00.00.68\_beta1

### Release Notes

This release is intended to be used on:

- TI816X (Linux)
- TI814X (Linux and Bios)
- OMAP3530 (Linux)
- OMAPL1XX (Linux)

## Introduction

This document is the Release Notes document for SYS/Link.

SYS/Link, also referred to as SysLink, is a software platform that simplifies the development of embedded applications in which a general-purpose microprocessor (GPP) controls and communicates with one or more processors (can be microprocessors / DSPs). SysLink provides control and communication paths between GPP OS threads and SYS/BIOS tasks. The SysLink product provides software connectivity between multiple processors. Each processor may run either an HLOS such as Linux, WinCE, Symbian etc. or a Real Time Operating System such as SYS/BIOS™ or PrOS. Based on specific characteristics of the Operating system running on the processor, the software architecture of each module shall differ. Current SysLink supports Linux on HLOS and SYS/BIOS on RTOS side. The associated Porting kit can be used to port SysLink to various OSes.

The SysLink product provides the following services to frameworks and applications:

- Processor Manager
- Inter-Processor Communication
- Utility modules

SYS/BIOS operating system is expected to be running on the slaves with all of these platforms.

This is an early adopter release with limited validation. This release is provided to give an early look at SysLink to framework/application developers.

The SysLink implementation is currently in-progress. Design changes are currently being evaluated and implemented for Linux-side sources to move the implementation of most modules (except Notify) to user-space. Several advanced features such as Dynamic Memory Mapping, Dynamic System Memory Management, Slave Error Handling, Advanced Power Management are not currently implemented and are planned for future releases.

## Release Components

The SysLink release package contains following components:



Components	Build: Commit Id
tar.gz containing the SysLink installation package	syslink_02.00.00.68_beta1.tar.gz
Documentation	Refer to index.html in the release package.
Sample applications	Sample applications demonstrating usage of the modules are provided in the release package under ti/syslink/samples.

## Prerequisites

- A Linux Host machine with CodeSourcery GNU Toolchain for ARM installed and added in path is required for building the kernel module. Note that the builds are tested with 2009-Q1 version of toolchain.Refer to User Guide for more info on paths and environmental variables
- For OMAP3530, the following **iommu patch** should be applied over the kernel,this can be found in the directory: \$SYSLINK\_ROOT/ti/syslink/buildUtils/hlos/knl/

Please refer to the **Install Guide** for more details.

## Dependencies

### Generic

Components	Repository	Build: Release Tag
TeraTerm (or any other terminal emulation program)	Not available as part of SysLink release.	Latest

### OMAP3530

Components	Repository	Build: Release Tag
SYSBIOS™	[1]	6.31.04.27
XDC tools	[2]	3.20.08.88
IPC	[3]	1.22.04.25
DSP CG Tools	[4]	c6000_7.2.0B1
CCS	[5]	version 4.2.07000 or later
Code Sourcery Tool Chain	[6]	arm2009q1-203
PSP release	[7]	AM35x-OMAP35x-PSP-SDK-03.00.00.03

- For details of how to set the different profile modes, see upgrade and compability information in this relese notes.

### TI81XX



Components	Repository	Build: Release Tag
SYSBIOS™	[8]	6.31.04.27
XDC tools	[2]	3.20.08.88
IPC	[3]	1.22.04.25
DSP CG Tools	[4]	c6000_7.2.0B1
VIDEO M3 and DSS M3 CG Tools	[4]	ELF: 4.6.4
CCS for TI816X	[5]	version CCS_5.0.1.00026
CCS for TI814X	[5]	version CCS_5.0.1.00026
Code Sourcery Tool Chain	[6]	arm2009q1-203
Linux PSP release for TI816X EVM	[9]	TI816X-LINUX-PSP-04.00.00.10
U-Boot for TI816X EVM	[9]	TI816X-LINUX-PSP-04.00.00.10
Linux PSP release for TI814X EVM	[10]	TI814X-LINUX-PSP-04.01.00.03
U-Boot for TI814X EVM	[10]	TI814X-LINUX-PSP-04.01.00.03
Framework components for Tiler	[11]	framework_components_3_21_00_15_eng

## OMAPL1xx

Components	Repository	Build: Release Tag
SYSBIOS™	[1]	6.31.04.27
XDC tools	[2]	3.20.08.88
IPC	[3]	1.22.04.25
DSP CG Tools	[4]	c6000_7.2.0B1
CCS	[5]	version 4.2.07000 or higher
Code Sourcery Tool Chain	[12]	arm2009q1-203
PSP release	[13]	DaVinci-PSP-SDK-03.20.00.11

## Documentation

- ReleaseNotes - This document.
- InstallGuide - Provides information to get started with installing SysLink and its dependencies, building SysLink and its sample applications, and executing the sample applications
- UserGuide - Provides information about SysLink architecture, modules, and enables users to write and debug applications using SysLink.
- DataSheet - Provides performance and resource usage information about SysLink.
- Doxygen generated API reference - Provides API reference documentation for SysLink modules. For API reference documentation of IPC modules, refer to the doxygen generated API reference in the IPC product



## What's Supported

### Features

#### Generic features

Syslink provides user and kernel side APIs for:

- Processor Manager
- Inter-Processor Communication protocols

In this engineering release, following HLOS modules are supported:

- ProcMgr
- Ipc
- Notify
- MessageQ
- ListMP
- SharedRegion
- MultiProc
- GateMP
- NameServer
- HeapBufMP
- HeapMemMP
- FrameQ
- RingIO

In this engineering release, following RTOS modules are supported:

- FrameQ
- RingIO

#### Ipc

The Ipc module initializes all IPC and SysLink components on the behalf of the application. It provides a single platform-specific location to implement system level module initialization and configuration. It also allows application to seamlessly configure the SysLink and IPC components.

Ipc module on multi-processors system resides on each processor and enables synchronizing between any two cores using the Ipc module for setting up the system configuration.

The Ipc is an optional module. The application writer can choose to use or not use Ipc module for configuring the modules. If not used, all IPC modules must be individually configured by the application writer. However, note that due to the complexity of this configuration, it is advisable to use the Ipc module for system configuration and synchronization.

---



### **Processor Manager module**

The Processor Manager on a master processor provides control functionality for a slave device. The ProcMgr module provides the following services for the slave processor:

- Slave processor boot-loading
- Read from or write to slave processor memory
- Slave processor power management
- Slave processor error handling

The Device Manager (Processor module) has interfaces for:

- Loader: There may be multiple implementations of the Loader interface within a single Processor instance. For example, COFF, ELF, dynamic loader, custom types of loaders may be written and plugged in.
- Power Manager: The Power Manager implementation can be a separate module that is plugged into the Processor module. This allows the Processor code to remain generic, and the Power Manager may be written and maintained either by a separate team, or by customer.
- Processor: The implementation of this interface provides all other functionality for the slave processor, including setup and initialization of the Processor module, management of slave processor MMU (if available), functions to write to and read from slave memory etc. All processors in the system shall be identified by unique processor ID. The management of this processor ID is done by the MultiProc module.

### **Inter Processor Communication protocols**

#### **Notify**

The Notify Manager manages the multiplexing/de-multiplexing of software interrupts over hardware interrupts. The Notify module uses notify drivers that do the actual management of the callback functions and interface to the hardware

#### **MessageQ**

The MessageQ module supports the structured sending and receiving of variable length messages. This module can be used for homogeneous or heterogeneous multi-processor messaging.

MessageQ provides more sophisticated messaging than other modules. It is typically used for complex situations such as multi-processor messaging. The following are key features of the MessageQ module:

- Writers and readers can be relocated to another processor with no runtime code changes.
  - Timeouts are allowed when receiving messages.
  - Readers can determine the writer and reply back.
  - Messages can reside on any message queue.
  - Supports zero-copy transfers.
  - Notification mechanism is specified by application.
-



## **GateMP**

This module provides the design for implementing multi-processor critical section gates with remote and local protection. The type of gate used, whether s/w (GatePeterson) or h/w (GateHwSpinlock) depends on the device capabilities.

## **ListMP**

The ListMP Manager is a doubly linked-list based module designed to be used in a multi-processor environment. It provides a way for multiple processors to create, access, and manipulate a link list in shared memory.

## **HeapBufMP/HeapMemMP**

The HeapBufMP memory manager provides functions to allocate and free storage from a heap of type HeapBufMP which inherits from IHeap. HeapBufMP manages a single fixed-size buffer, split into equally sized allocable blocks. The HeapBufMP module is intended as a very fast memory manager which can only allocate blocks of a single size. It is ideal for managing a heap that is only used for allocating a single type of object, or for objects that have very similar sizes.

The HeapMemMP module is a variable size multi-processor memory manager, protected with a multi-processor GateMP.

## **FrameQ**

FrameQ module provides queue based data transfer mechanism between clients. It allows inserting and retrieving frame nodes from the queue by clients. Each node contains a pointer to a data buffer (frame buffer), implementation specific information and also carries additional information pertaining to the frame buffer.

## **RingIO**

The RingIO component provides data streaming based inter-processor communication between GPP and DSP using ring buffer data as the transport.

## **Utility Services Provided**

### **SharedRegion**

The SharedRegion module is designed to be used in a multi-processor environment where there are memory regions that are shared and accessed across different processors.

This module creates a shared memory region lookup table. This lookup table contains the processor's view for every shared region in the system. Each processor has its own lookup table. Each processor's view of a particular shared memory region can be determined by the same table index across all lookup tables. Each table entry is a base and length pair. During runtime, this table along with the shared region pointer is used to do a quick address translation.

If specified in configuration, a multi-processor heap can be created in the SharedRegion, which can be used by applications for their shared memory requirements. The heap created is of type HeapMemMP and is protected by the system gate.

---



## List

The List module makes available a set of functions that manipulate List objects accessed through handles of type List\_Handle. Each List contains a linked sequence of zero or more elements referenced through variables of type List\_Elem, which are typically embedded as the first field within a structure.

## MultiProc

Many multi-processor modules have the concept of processor id. MultiProc centralizes the processor id management into one module.

Generally to improve performance and to minimize data footprint, MultiProc provides information about the remote processors.

## NameServer

The NameServer module manages local name/value pairs that enable an application and other modules to store and retrieve values based on a name. The module supports different lengths of values. The add/get functions are for variable length values. The NameServer module can be used in a multiprocessor system. The module communicates to other processors via the Remote driver.

## Sample applications

Sample applications to demonstrate usage of modules have been provided for:

- ProcMgr
- Notify
- MessageQ
- HeapBufMP
- HeapMemMP
- ListMP
- SharedRegion
- FrameQ
- RingIO
- GateMP

*The ProcMgr sample application can also be used as an external loader before running the sample applications for other modules.*

## What's not supported

- Unsupported features in this release:
  - TI816X simulator
  - HeapMultiBufMP module

## Host Support

This release has been validated on the following host machines:

- Red Hat Enterprise Linux 4 for Code Sourcery toolchain - 2009Q1 -203 version
- Windows XP SP2 for CCS v4 Installation



## Device Support

This release supports the following devices.

- TI81XX
- OMAP3530
- OMAPL1XX

## Validation Information

This engineering release has been validated using the following configurations:

Platform	HLOS OS distribution	HLOS Tool-chain	HLOS Base-port	RTOS OS distribution	RTOS Tool-chain	IPC version	XDC version
TI816X	Linux kernel 2.6.37-rc3	CodeSourcery arm-2009q1-203	TI816X-LINUX-PSP-04.00.00.10	6.31.04.27	CGTOOLS v7.2.0B1, arm_4.6.4	1.22.04.25	3.20.08.88
TI814X	Linux kernel 2.6.37-rc3	CodeSourcery arm-2009q1-203	TI814X-LINUX-PSP-04.01.00.03	6.31.04.27	CGTOOLS v7.2.0B1, arm_4.6.4	1.22.04.25	3.20.08.88
OMAP3530	Linux kernel 2.6.32-rc7	CodeSourcery arm-2009q1-203	AM35x-OMAP35x-PSP-SDK-03.00.00.03	6.31.04.27	CGTOOLS v7.2.0B1	1.22.04.25	3.20.08.88

NOTE: This release has also been validated with OMAP3530 Kernel 2.6.37-rc7, with an early code drop from Linux PSP. This is to enable adaptation of kernel 2.6.37 when the actual release package for OMAP3530 with this kernel is available

## New in this release

### Generic Defect Fixes

Defect ID	Headline	Fix Information
SDOCM00079310	install guide documenation is not adequate to run syslink user side examples on ARM+DSP only devices	The SysLink TI81XX Install Guide has been updated to add a separate section for EZSDK build and run for ARM+DSP usage.
SDOCM00079216	[RingIO] Reader and writer client structures in opened object are corrupted	Cache_inv call is needed for reader and write client structures of newly created object in RingIO_openByAddr to get valid values.
SDOCM00078898	[FrameQ] Memory leak is observed in FrameQ create and open APIs	Rtos side sources are updated to fix the memory leak issue .
SDOCM00078861	TransportShm notifyEventId is present in both the module config as well as instance params	notifyEventId field has been removed from instance params for TransportShm module.
SDOCM00078774	[FrameQ] Basic frame transfer operation from ARM to DSP fails in Netra EVM	FrameQBufMgr_open call implementation is updated to do invalidation of pnameserver entry before opening syslinkMemMgr
SDOCM00078635	[SYSLINK_BUILD] SyslinkSamples intermediate files are not getting clean while kernel sample build cleaning.	ti/syslink/samples/hlos/common path was not included while kernel build cleaning. So included this path in all kernel sample Makefile.
SDOCM00078577	SysLink should ship with pre-built libraries on RTOS	Productization is updated to ship the libraries in release.
SDOCM00078553	[FrameQ] FrameQ_put raises an exception when writer starts putting frames before reader opens the frameQ.	Sources are updated to ensure that FrameQ_put from writer side passes even there is no reader opened.



SDOCM00078409	Addresses hard coded in Notify sample must be removed	Hardcoding of shared addresses like NOTIFY_SHARED MEM has been removed from sample app.
SDOCM00078248	[Notify] Notify user-side sample application sometimes hangs on TI816X on second run	The Linux-side implementation of interrupt handling code was corrected to resolve interrupt miss race conditions, and a SysBIOS release was received that resolves the GateDualCore issue.
SDOCM00078178	SysLink device node names has '!' instead of '/'	Sources are updated and now dev nodes of syslink are getting created directly under /dev/. Device nodes of syliink are in the form /dev/syslinkipc_xxxx where xxxx is the syslink module name.
SDOCM00077356	[SAMPLEAPP] RingIO user side sample application hangs on TI816X if executed after frameq/messageq sample	Issue is no longer observed
SDOCM00076852	[RingIO] No error message is returned when RingIO instance is created twice with same name.	RingIO asserts when it created twice with same name and returns valid status code.
SDOCM00076844	[RingIO] GateMP assertions are observed in RTOS side when RingIO created+deleted is performed for more than 64 iterations	Issue is no longer observed
SDOCM00076842	[RingIO] Nameserver assertions are observed when RingIO instance is created second time in RTOS side.	Issue is no longer observed
SDOCM00076878	[RingIO] ringIO sample shows assertions when run alternatively with heapmemmp sample	Issue is no longer observed
SDOCM00074925	[SAMPLE APP] Assertions seen in RingIO_gpp when run in a script along with other samples.	Issue is no longer observed
SDOCM00077183	[RINGIOGPP] RingioGpp kernel side sample application hangs on TI814x	Issue is no longer observed
SDOCM00076495	RingIO single core app fails in RingIO_Open on dsp of Centaurus	Issue is no longer observed
SDOCM00076534	[Sample app] Remove the user interference during the kernel sample app execution	The ProcMgrApp has been updated to allow either only starting up, or shutting down the slave core. This enables scripting for kernel samples and the Ctrl Z method is not required.
SDOCM00075790	[RingIO] Code cleanup is required in RingIO sample application on both HLOS and RTOS side.	RingIO code has been cleaned up and is readable and robust now. It is fixed as part of non-RTSCization and restructuring of RingIO. RingIO now has common code between HLOS and RTOS
SDOCM00074923	[SAMPLE APP] RingIO sample application crashes in second consecutive run	Issue is no longer observed
SDOCM00069690	[Notify] Event loss may be seen if notifications are sent while events are being registered/unregistered on remote processor	The NotifyDriverShm implementation has been updated to ensure that it behaves correctly when register/unregister are done while events are being sent from the other processor.
SDOCM00069582	[FrameQ] Unloading of FrameQ knl side sample app module sometimes hang on the first run	Issue is not seen any more in latest code base
SDOCM00068041	[FrameQ] FrameQ kernel side sample application fails after multiple execution	This issue is not observed in not seen in latest code base.



## Platform-specific Defect Fixes

### OMAP3530

Defect ID	Headline	Release Note
-----------	----------	--------------

### TI81XX

Defect ID	Headline	Release Note
-----------	----------	--------------

### OMAPL1XX

Defect ID	Headline	Release Note
-----------	----------	--------------

## Operating System specific Defect Fixes

### Linux

Defect ID	Headline	Release Note
-----------	----------	--------------

## Upgrade and compatibility information

### Changes from last release 02.00.00.67

#### Versions of dependencies changed

- XDC version updated to 3.20.08.88
- IPC version updated to 1.22.04.25,
- SYS/BIOS version updated to bios 6.31.04.27
- Linux PSP update for TI816x updated to 04.00.00.10
- Linux PSP update for TI814x updated to 04.01.00.03
- Code generations tools updated to 4.6.4 for M3 cores on ti816x and ti814x
- Frame work components updated to 3.21.00.15

#### Additional build flags needed

- On HLOS side you must define SYSLINK\_BUILD\_HLOS for HLOS build to succeed in user side and kernel side base Makefile.inc

```
COMPILE_FLAGS += -DSYSLINK_BUILD_HLOS
```

- On RTOS side you must define following flags in your config.bld for RTOS build to succeed. e.g. For building syslink rtos side for m3 elf please add following compiler flags to config.bld

```
M3_ELF.ccOpts.suffix += "-g --gcc -s -DSYSLINK_BUILD_RTOS";
```

- These defines are needed because now RingIO has a common code between HLOS and RTOS.



### Changes for RingIO/FrameQ applications

- RingIO/FrameQ applications on rtos need to call `SysLink_setup()` before doing `Ipc_attach()`. Please refer rtos side ringIO applications

```
include <ti/syslink/SysLink.h> .... tsk_fxn() { .... SysLink_setup();
Ipc_attach(procId); .... }
```

### Changes for Specific for RingIO applications

- Source files (\*.c) for RingIO and ClientNotifyMgr have been moved to `$(SYSLINK_ROOT)/ti/syslink` directory as it is the common code between rtos and hlos
- There are minor changes in RingIO API signatures. Please refer `$(SYSLINK_ROOT)/ti/syslink/RingIO.h` for details. New API signatures are as followed

```
RingIO_Handle RingIO_create (const Void * params);
Void RingIO_Params_init (Void * params);
UInt32 RingIO_sharedMemReq (const Void * params,
```

```
RingIO_sharedMemReqDetails *
sharedMemReqDetails);
```

- RingIO\_Params has only one field 'name'. All parameters essential for creation of RingIO have been moved to RingIOShm\_Params as RingIOShm is the new shared memory implementation of RingIO and these parameters refer to instance of RingIOShm. Please refer corresponding header files i.e. RingIO.h and RingIOShm.h for details.
- Please note that RingIOTransportShm does not exist now. All RingIO applications have to include RingIOShm.h to use RingIO and RingIO creation sequence has changed Please refer RingIO sample application for details.

```
include <ti/syslink/RingIOShm.h>
```

- RingIO has 'classic C' based configuration for which you can refer to `$(SYSLINK_ROOT)/ti/syslink/cfg/syslinkCfg.c`, RingIO module configuration is done through this C file.
- While moving RingIO to classic C it is assumed that Application's cfg will include `xdc.useModule()` calls for all xdc based dependencies e.g ipc and bios modules.
- Since RingIO and ClientNotifyMgr are no longer XDC modules, you have to remove following code for existing RingIO applications's cfg:

```
var ClientNotifyMgr = xdc.useModule ('ti.syslink.ipc.rtos.ClientNotifyMgr') ;
var RingIOTransportShm = xdc.useModule ('ti.syslink.ipc.rtos.RingIOTransportShm') ;
var RingIO = xdc.useModule ('ti.syslink.ipc.rtos.RingIO') ;
```

### Device node creation changes

- Now SysLink creates device nodes under `/dev` directory directly. Device nodes of SysLink are in the form `/dev/syslinkipc_xxxx` where `xxxx` is the SysLink module name.

### ProcMgr All Boot Mode Support Addition

- There are 4 boot modes support enabled for ProcMgr module.
  - ProcMgr\_BootMode\_Boot
  - ProcMgr\_BootMode\_NoLoad\_Pwr
  - ProcMgr\_BootMode\_NoLoad\_NoPwr
  - ProcMgr\_BootMode\_NoBoot



NOTE: Refer to **User Guide** for more info about different boot modes.

### ProcMgr Sample Application Configurable

ProcMgr sample application has added one more configurable option as boot mode.

- This boot mode option has been added as last argument to maintain the backward compatibility.
- This argument is optional input as boot mode. The default boot mode for ProcMgr is **ProcMgr\_BootMode\_Boot**.
- The address of symbol `_Ipc_ResetVector` in the slave executable has been muxed with absolute path of slave executable argument in ProcMgr sample application. The `_Ipc_ResetVector` symbol address needs to be provided to the `procmgrapp` if any boot mode other than `ProcMgr_BootMode_Boot` is to be used.
- The usage of ProcMgr sample application is:

```
procmgrapp <processor ID>
                <{_Ipc_ResetVector symbol address} or {absolute
path of slave executable}>
                [Optional: Startup proc only - 0, Shutdown proc only
- 1]
                [Optional: Boot mode - 0, NoLoad_Pwr mode - 1,
NoLoad_NoPwr mode - 2, NoBoot mode - 3]
```

- The limitation is that if boot mode argument is used then optional argument start/shutdown flag becomes mandatory.
- Example for `ProcMgr_BootMode_Boot` mode:

```
./procmgrapp_debug 0 frameq_ti81xx_dsp.xe674
```

NOTE: The argument[1] is procID of slave core and argument[2] is absolute path of slave executable.

- Example for `ProcMgr_BootMode_NoLoad_Pwr` mode:

```
./procmgrapp_debug 0 0x8b035000 0 1
./procmgrapp_debug 0 0x8b035000 1 1
```

NOTE: The argument[1] is procID of slave core, argument[2] is `_Ipc_ResetVector` symbol address, argument[3] startup flag and argument[4] is boot mode.

- Example for `ProcMgr_BootMode_NoLoad_NoPwr` mode:

```
./procmgrapp_debug 0 0x8b035000 0 2
./procmgrapp_debug 0 0x8b035000 1 2
```

NOTE: The argument[1] is procID of slave core, argument[2] is `_Ipc_ResetVector` symbol address, argument[3] startup flag and argument[4] is boot mode.

- Example for `ProcMgr_BootMode_NoBoot` mode:

```
./procmgrapp_debug 0 0x8b035000 0 3
./procmgrapp_debug 0 0x8b035000 1 3
```

NOTE: The argument[1] is procID of slave core, argument[2] is `_Ipc_ResetVector` symbol address, argument[3] startup flag and argument[4] is boot mode.



**Misc changes**

- For TI816x, now we use the tiler driver implementation provided in kernel of PSP release.

**Changes from last release 02.00.00.66****Versions of dependencies changed**

- XDC version update to 3.20.07.86
- IPC version update to 1.22.03.23,
- SYS/BIOS version update to bios 6.31.03.25
- Linux kernel version update to 2.6.37 rc3 for TI814x

NOTE: This release has also been validated with OMAP3530 Kernel 2.6.37-rc7, with an early code drop from Linux PSP. This is to enable adaptation of kernel 2.6.37 when the actual release package for OMAP3530 with this kernel is available

**Build changes**

- The default build option now builds for TI81XX instead of OMAP3530 for both kernel and user-side of SysLink and sample applications
  - SYSLINK\_PLATFORM=TI81XX no longer needs to be provided to build for TI81XX
  - If SYSLINK\_VARIANT is not provided, default build is for TI816X
  - SYSLINK\_VARIANT=TI814X needs to be provided to build for TI814X
  - SYSLINK\_PLATFORM=OMAP3530 needs to be provided to build for OMAP3530
- SysLink sample application user-side executables are now generated with profile suffix
  - Debug binaries are generated with `_debug`
  - Release binaries are generated with `_release`
- SysLink samples platform configuration changed to use deviceName as TMS320TI814X
- User-side build system has been updated to remove 'standard' build
  - If just `make` command is given, it builds for both 'debug' and 'release'
  - To build only for 'debug', use command `make debug`
  - To build only for 'release', use command `make release`
- The libraries (syslink.a\_debug and syslink.a\_release) are generated
  - Generation of syslink.a is no more supported
- For TI81XX, sample applications on both sides i.e. HLOS and RTOS can now be built for EZSDK
  - If built for EZSDK, all sample applications run only between the Cortex A8 and DSP cores.
  - To build for EZSDK, provide SYSLINK\_SDK=EZSDK as an additional make option while building both kernel-side and user-side HLOS sample applications. An example kernel-side sample application build for EZSDK is shown below:

```
$ cd $SYSLINK_ROOT/ti/syslink/samples/hlos/messageQ/knl/Linux
$ make ARCH=arm
CROSS_COMPILE=/db/psp_git/syslink_toolchains/arm-2009q1/bin/arm-none-linux-gnueabi-
SYSLINK_SDK=EZSDK
```

An example user-side sample application build for EZSDK is shown below:

```
$ cd $SYSLINK_ROOT/ti/syslink/samples/hlos/messageQ/usr/Linux
$ make SYSLINK_PLATFORM=TI81XX SYSLINK_SDK=EZSDK
```



- To build RTOS side library and samples for EZSDK the build command has been changed for example building in debug profile for EZSDK

```
$ export XDCARGS="profile=debug SDK=EZSDK"
$ xdc all XDCBUILDCFG="$SYSLINK_ROOT/config.bld" -PR .
```

For further details, please refer to the Install Guide.

### Sample applications configurable

Sample applications are now configurable to run for any combination of slave cores

- An additional build step is required for all HLOS user-side sample applications to build the common syslinksamples library:

```
$ cd $SYSLINK_ROOT/ti/syslink/samples/hlos/common/usr/Linux
$ make
```

If the sample application is to be run with any one or two slave cores, the appropriate parameter can be passed to the application

For user application, this can be done by passing the appropriate first argument for number of slave cores and corresponding slave executable paths. For example, to run messageqapp user-side sample with only two cores (say core 0 and core 2):

```
$ ./messageqapp_debug 2 0 ./messageq_ti81xx_dsp.xe674 2
./messageq_ti81xx_vpssm3.xem3
```

To run messageqapp user-side sample with only one core (say core 1):

```
$ ./messageqapp_debug 1 1 ./messageq_ti81xx_videom3.xem3
```

For kernel application, this can be done by passing the appropriate values for NUMPROCS and PROCID module params. For example, to run messageqapp kernel-side sample with only two cores (say core 0 and core 2):

```
$ ./procmgrapp_debug 0 ./messageq_ti81xx_dsp.xe674 0
$ ./procmgrapp_debug 2 ./messageq_ti81xx_vpssm3.xem3 0
$ insmod messageqapp.ko NUMPROCS=2 PROCID=02
$ rmmod messageqapp
$ ./procmgrapp_debug 0 ./messageq_ti81xx_dsp.xe674 1
$ ./procmgrapp_debug 2 ./messageq_ti81xx_vpssm3.xem3 1
```

To run messageqapp kernel-side sample with only one core (say core 1):

```
$ ./procmgrapp_debug 1 ./messageq_ti81xx_videom3.xem3 0
$ insmod messageqapp.ko NUMPROCS=1 PROCID=1
$ rmmod messageqapp
$ ./procmgrapp_debug 1 ./messageq_ti81xx_videom3.xem3 1
```



**ProcMgr sample application feature addition**

ProcMgr sample application can now be optionally invoked only to startup or shutdown a slave core

- With this feature, the Ctrl Z method of running sample applications is no longer required, and hence is not documented.
- If invoked with default parameters, it both starts up and shuts down the slave core
  - If an additional final parameter is provided as 0, it only starts up the slave core and exits.
  - If the additional final parameter is provided as 1, it only shuts down the slave core and exits.

For example, to only start up the slave core:

```
$ ./procmgrapp_debug 0 ./messageq_ti81xx_dsp.xe674 0
```

To only shutdown the slave core:

```
$ ./procmgrapp_debug 0 ./messageq_ti81xx_dsp.xe674 1
```

To both startup and shutdown the slave core:

```
$ ./procmgrapp_debug 0 ./messageq_ti81xx_dsp.xe674
```

**Shared memory section name change**

- In earlier releases there was hard coding of addresses in internal samples files. This did not allow the samples to be configurable to any memory map. To change that, the shared region base address and length of shared regions for all sample applications are derived from the values passed in SysLink configuration files. The system integration needs to pass the shared memory addresses using fixed names SR0 and SR1 and the samples will auto configure themselves based on the values passed.
  - TI814x - custom platforms in \$SYSLINK\_ROOT/ti/syslink/samples/rtos/platforms
  - OMAP3530 - config.bld
- Shared memory sections name are renamed to SR0 and SR1 in custom platforms and config.bld.
  - SHARED -> SR0
  - APPLICATION -> SR1

**Work-around for Four core sample app bug on TI814x**

- *Four core FrameQ sample:*
- *Step1:*

Go to \$(SYSLINK\_ROOT)/ti/syslink/samples/rtos/platforms/ti814x/arm/Platform.xdc - Replace "EXT\_RAM" by "DDR2" everywhere.

- *Step2:*

```
** Go to
$(SYSLINK_ROOT)/ti/syslink/samples/rtos/ti81xx/frameq_ti814x/FrameQ_ti814x_arm.cfg
```

Replace following lines:

```
/* Make shared memory cacheable */
Mmu.setFirstLevelDescMeta(0x8FB00000, 0x8FB00000, attrs);

/* make code space cacheable */
Mmu.setFirstLevelDescMeta(0x8F700000, 0x8F700000, attrs);
```

by



```

/* Make shared memory cacheable */
Mmu.setFirstLevelDescMeta(0x8E000000, 0x8E000000, attrs);

/* make code space cacheable */
Mmu.setFirstLevelDescMeta(0x8D000000, 0x8D000000, attrs);

        // Set the descriptor for each entry in the address range
for (var i=0x80000000; i < 0x0A600000; i = i + 0x00100000) {
    // Each 'SECTION' descriptor entry spans a 1MB address range
    Mmu.setFirstLevelDescMeta(i, i, attrs);
}

var memmap = Program.cpu.memoryMap;
var DDR = null;

// Find DDR in memory map
for (var i=0; i < memmap.length; i++) {
    if (memmap[i].name == "DDR2") {
        DDR = memmap[i];
    }
}

// Place the MMU table in the DDR memory segment if it exists
if (DDR != null) {
    var sectionName = "ti.sysbios.family.arm.a8.mmuTableSection";
    Program.sectMap[sectionName] = new Program.SectionSpec();
    Program.sectMap[sectionName].type = "NOINIT";
    Program.sectMap[sectionName].loadSegment = "DDR2";
}
else {
    print("No DDR memory segment was found");
}

```

- *Step 3:* Rebuild ti/syslink/samples/rtos/platforms.
- *Step 4:* Rebuild four core samples.
- *Four core RingIO sample:*

```

** Follow Step 1 mentioned above.
** Go to
$(SYSLINK_ROOT)/ti/syslink/samples/rtos/ti81xx/ringIO_ti814x/RingIO_ti814x_arm.cfg
and follow Step 2.
** Follow Step 3 and 4.

```



### Miscellaneous changes

Script files have been provided for all combination of sample application runs for debug and release builds for the TI81XX platform.

- The script files are provided in tools/scripts folder in the release package
- To ensure there is no hard coding of address, some changes in sample applications have been made to pass the address used by the sample. For e.g. SharedRegion sample has been updated to ensure that the first argument is memory address used.

*Note:* Issue SDOCM00078248 mentioned in known issues section has been root-caused to a SysBios issue. To get a fix for this issue, users can move to an upcoming BIOS release.

### Changes from last release 02.00.00.65

- Versions of dependencies changed - moved to xdc 3.20.05.76, ipc 1.22.00.19, bios 6.31.00.18
- Disable CONFIG\_PROVE\_LOCKING option in kernel build (for ti816x and ti814x) through make menuconfig. Go to Kernel hacking menu and disable "Lock debugging: prove locking correctness" option. This is required to avoid lockdep warnings regarding mutex locks that shows up during sample application execution.
- **Build system changes** - Build system for HLOS side has been modified to
  - build for various profiles such as standard, debug and release
  - executable for different platforms coexist in corresponding platform folder under ti/syslink/bin
  - feature to clean based on profile/platform

For more details refer to following section in UserGuide: HLOS build system changes

- IPC module validates shared region numentries - It validates SharedRegion configuration of slave side against that of Master; user applications must add the following lines to their config file(.cfg on RTOS)

```
var Syslink = xdc.useModule ('ti.syslink.ipc.rtos.Syslink');
```

*Note:*

- There are two issues with current implementation:
  1. This module plugs in Syslink\_attach in to Ipc.userFxn.attach. When any other user attempts to plug in their own Ipc.UserFxn for attach or detach, it results in the SysLink UserFxn not getting called. This happens because SysLink (in Syslink.xs), sets the value of Ipc.numUserFxn to 1, instead of incrementing it. Hence, if the user had incremented the value earlier to include their UserFxn as well, this overwrites the incremented value and resets it. So to use the SysLink function user should not plugin any other functions to Ipc.userFxn.
  2. Syslink module depends on RingIOTransportShm which internally creates ClientNotifyMgr task, this dependency will be removed in future versions.
- Ipc\_detach() usage - On any processor, Ipc\_detach() with the owner processor of SharedRegion 0 should be called only once the processor is detached from all other processors
- GateMutex\_enter() assertion - solution - BIOS has added an assertion in GateMutex\_enter() call which is raised when any function calls GateMutex\_enter() from an ISR context(SWI/HWI); for example calling System\_printf() from Notify callback functions; applications can still use System\_printf() in callbacks if SysMin module is used instead of SysStd for printf function; this can be done by loading SysMin module by adding following lines to app config file

```
SysMin = xdc.useModule('xdc.runtime.SysMin');
System = xdc.useModule('xdc.runtime.System');
System.SupportProxy = SysMin;
```



*Note: When SysMin is used, prints on RTOS (slaves) cannot be seen during execution, they can only be seen using ROV once the execution is complete/ exited*

- **Linux PSP:** Linux PSP version for ti816x is based on Kernel 2.6.37, There are certain changes to be done in file system and boot args to get it working.

```

** NFS chnages
- Update console from ttySx to ttyOx in /etc/inittab file
- Rename the file /etc/udhcpc.d/50default to something else, also
rename the directory /etc/udhcpc.d to something else.
** Boot args
- While passing the bootargs pass "console=ttyO2"
- on the terminal Set the variable 'machid' to af0

```

*NOTE: These are the changes that we made to get it work for us, but these are not exhaustive, if your setup doesn't work even after doing these changes please refer Linux PSP documentation for details or contact PSP team.*

- SYSLINK\_DUCATI\_ENABLE - Ducati mmu is enabled by default, user need not use "-DSYSLINK\_DUCATI\_ENABLE"
- BIOS Timer configuration - BIOS timer for DSP needs to be configured to run at 32Khz to match with the default Linux timer configuration, default timer on BIOS runs at 27Mhz. This can be done by adding following lines to DSP config (.cfg) of the application. If this configuration is not done Task\_sleep() hangs on DSP core of Ti81XX.

```

var Timer      = xdc.useModule('ti.sysbios.timers.dmtimer.Timer');
Timer.intFreq.hi = 0;
Timer.intFreq.lo = 32768;

```

- Missing Notification issue: Notify\_sendEvent on HLOS(Linux) has been updated so that the interrupts sent from HLOS are not missed

#### Changes from last release 02.00.00.64

- Added support for TI814X for both combinations A8+bios and A8+Linux
- Single porting kit release supporting TI816X, TI814X, Omap3530 and Omapl1xx
- Build command for hlos side has been changed to support TI814X and TI816X together in one release. SYSLINK\_VARIANT added to build command which is still optional for TI816X users.
- RTOS Samples(samples/rtos) directory has been rearranged to put inter ducati and single core apps in to ti81xx directory inside samples/rtos directory.
- Bug fixes.

#### Changes from last release 02.00.00.54

- MessageQ module's numHeaps default value has been changed to 8 as part of IPC 1.22.00.10 support.
- Now L1 & L2 caches are configured in SysLink custom platform files for DSP of T18XX. See <SYSLINK\_ROOT>ti/syslink/samples/rtos/platforms/ti816x/dsp/platform.xdc
- TI814X support removed
- Move to xdc 3.20.05.69, ipc 1.22.00.13\_eng and bios 6.31.00.06\_eng



**Changes from last release 02.00.00.53**

- Supports BIOS 6.31.00.03\_eng and IPC 1.22.00.04\_eng
- Includes TI814X port but not validated on TI814X evm

**Changes from last release 02.00.00.52**

- Supports BIOS 6.30.03.43 and IPC 1.21.03.24
- Reserved notify event Ids have been serialized (FrameQBufMgr-0,FrameQ-1,TransPortShm-2,RingIO-3,NameServerRemoteNotify-4)
- Processor Names have been modified (DSS to VPSS-M3 , VIDEO to VIDEO-M3)
- MultiProc\_getId (NULL) API is no longer supported.Applications has to use MultiProc\_self() API to know the local processor id.
- Ipc\_detach now synchronizes using shared memory.
- linkcmd.xdt is added in sysLink/ipc/rtos, which will contain the command to align the entry point to 0x400 as needed for the DSP. Correspondingly changes are done in Linux-side to directly use the entry point.So user needs to add the following to their rtos side cfg file:

```
xdc.loadPackage ('ti.syslink.ipc.rtos');
```

- COFF format for M3 cores(VIDEO-M3 and VPSS-M3) is not supported.

**Changes from last release 02.00.00.51**

- Supports BIOS 6.30.02.35\_1\_eng.This bios release has ducati unicache fixes to improve the cache performance. Release is validated on this bios release for all OMAP3530,OMAPL1xx and DM8168.
- numNotifyEntries create time param is added to FrameQBufMgr\_ShMem\_Params. This denotes how many FrameQ (in turn FrameQ writers of different FQs) requires notification from the FrameQBufMgr.

(This is done to reduce control structure size of ClientNotifyMgr to a very minimum to avoid cache overhead). By default this value is set to 1.So if FrameQBufMgr is going to be used by only one FrameQ,no modification is required at application level. Otherwise, Applications has to pass this parameter based on the number of FrameQs using this FrameQBufMgr instance.

- Now FrameQ\_delete API frees up any remaining frames that are there in its queues so frames won't get lost and will be available in FrameQBufMgr.

**Changes from last release 02.00.00.50**

- Updated to support ELF format for DSP of DM8168. By default syslink builds and assumes elf format for all video,dss and dsp executables. if COFF format needs to be supported for DSP, pass SYSLINK\_LOADER=COFF to syslink kernel build.
- To use ELF format dsp executable for OMAPL1xx, pass SYSLINK\_LOADER=ELF to syslink kernel build.
- Support for TI816X-LINUX-PSP-04.00.00.05 PSP release for DM8168 is added.
- IvahdDefs.h header file removed from syslink package as syslink does not support IVAHDs. All ProcMgr applications will have to remove it if they have included it.



**Changes from last release 02.00.00.47**

- Updated to support IPC 1.21.02.17, BIOS 6.30.02.35 and XDC tools 6.20.01.42
- Code generation tools c6000\_7.0.0 is used for validation. Versions below 7.0.0 is not supported.
- kernel version 2.6.34 support is added for DM8168( TI816X).
- GateMP\_LocalProtect\_TASKLET, GateMP\_LocalProtect\_PROCESS, GateMP\_LocalProtect\_THREAD support is added in FrameQ and FrameQBufMgr.
- Samples tested on TI816X EVM/OMAP3530/Primus
- Impact to applications as TI816X part name is changed from DM740 to DM8168. All references to DM740 now become DM8168.
- Supports IPC 1.21.02.17
- Need to build ti/syslink/samples/hlos/procMgr/usr/Linux before other samples since other samples depend on ti/syslink/lib/samples/procmgrapp.o

**Changes from last release 02.00.00.46**

- Multiple load/runs of application using ProcMgr
- Supported on TI816X simulator
- GateMP\_LocalProtect\_TASKLET, GateMP\_LocalProtect\_PROCESS, GateMP\_LocalProtect\_THREAD support is added in FrameQ and FrameQBufMgr.
- Samples tested on TI816X EVM
- No Impact to applications
- Supports IPC 1.20.01.27

**Changes from last release 02.00.00.45**

- ProcMgr modified for stability issues
- ProcMgr calling clk APIs from kernel instead of touching actual registers for enabling clocks
- Samples tested on TI816X EVM
- FrameQ fixes for supporting Swi protection for FrameQ and FrameQBufMgr instances
- No Impact to applications

**Changes from last release 02.00.00.44**

- Major modules (ProcMgr, MessageQ, Notify, FrameQ, RingIO) working on TI816X EVM

**Changes from last release 02.00.00.43**

- No impact to applications

**Changes from last release 02.00.00.42**

- GateMp proxy settings from the rtos side sample applications removed.
- DM740 samples and interducati samples are verified BY enabling Ducati\_Cache\_config and Ducati\_Cache\_Enable.
  - To enable, Select video or dss in ccs, go to scripts in the menu item scripts->Cache Configuration
    - Set Cache configurations by clicking on Ducati\_Cache\_config
    - Enable Ducati Unicache by clicking on Ducati\_Cache\_Enable



**Changes from last release 02.00.00.41**

- Rtos side build command has been changed. Now by default the build profile is whole\_program\_debug. If you want to change the profile users can specify profiles(debug,release etc) through xdc command line build using XDCARGS. There are two ways to do this.
  - set environment variable XDCARGS with the desired pprofile and build syslink rtos side.

```
export XDCARGS=profile=debug
```

```
export XDCARGS=profile=release
```

- Pass XDCARGS in xdc build command it self.

```
xdc XDCARGS="profile=debug" -PR .
```

- User-side applications need to now include SysLink.h instead of UsrUtilsDrv.h:
  - #include <ti/syslink/SysLink.h>
  - UsrUtilsDrv\_setup API has been renamed to SysLink\_setup: This API must be called in every process using SysLink, before making any SysLink API calls.
  - UsrUtilsDrv\_destroy API has been renamed to SysLink\_destroy: This API must be called to cleanup SysLink in every process at the end, after completion of all SysLink API calls.
- User-side applications calling Ipc\_control APIs now need to now include ti/syslink/IpcHost.h instead of \_Ipc.h:
  - #include <ti/syslink/IpcHost.h>
- FrameQ/FrameQBufMgr instances now derieve cache flags from the shared region. Only cpuAccessFlags are excpted from the create/open API. To avoid cache overhead in FrameQ/FrameQBufMgr instances that are going to be used with in processor or between inter ducati, Applications need to specify a cache disabled shared region(i.e Shared region that has cacheFlag set to false).
- By default SysLink plugs in daefault gate types into three proxies of GateMP module. The supported gate proxies are

```
GateMP_RemoteProtect_SYSTEM
GateMP_RemoteProtect_CUSTOM1
GateMP_RemoteProtect_CUSTOM2
```

- The default settings for the GateMP module on OMAP 3530, OMAPL1XX are as follows:
  1. GatePeterson module has been configured with 512 instances.
  2. GatePeterson is plugged into GateMP as both GateMP\_RemoteProtect\_SYSTEM and GateMP\_RemoteProtect\_CUSTOM1. So applications can create the gate that uses GatePeterson as remote gate by specifying any one of these protection levels (For details see GateMP\_RemoteProtect Enum in GateMP.h of IPC package).
  3. If the remote gates are not sufficient, applications need to increase the number of instances configured in GatePeterson module both in HLOS side and RTOS side Gatepeterson module.
    1. On HLOS side this can be done by changing the configuration of GatePeterson module in Platform.c file (\$SYSLINK\_ROOT\ti\syslink\family\hlos\kn\omap3530\Platform.c). Refer to the User Guide for more details. For example, see below:

```
var GatePeterson      = xdc.module('ti.sdo.ipc.gates.GatePeterson');
GatePeterson.numInstances = 580;
```

1. No gate is plugged in for GateMP\_RemoteProtect\_CUSTOM2.



2. To change the gate types that are associated with three remote protection levels, application need to rebuild syslink

1. By doing changes in the file GateMPDefs.h in \$\$SYSLINK\_ROOT/ti/syslink/inc directory of SysLink.
2. On RTOS side this change has to be done in application's cfg file. For details refer to any syslink examples.

The following code shows how to plugin GatePeterson as RemoteCustom1Proxy in GateMP module to associate GateMP\_RemoteProtect\_CUSTOM1 with GatePeterson.

```
var GatePeterson      = xdc.module('ti.sdo.ipc.gates.GatePeterson');
var GateMP            = xdc.useModule ('ti.sdo.ipc.GateMP');
GateMP.RemoteCustom1Proxy = GatePeterson;
```

- The default settings for the GateMP module on TI816X(DM740)are as follows:

1. GateHWSpinlock module has been configured with 64 instances(locks).
2. GateHWSpinlock is plugged into GateMP as GateMP\_RemoteProtect\_SYSTEM. So applications can create the gate that uses GateHWSpinlock as remote gate by specifying GateMP\_RemoteProtect\_SYSTEM protection level during instance creation. (For details see GateMP\_RemoteProtect Enum in GateMP.h of IPC package).
3. The number of GateHWSpinlocks are limited to 64 in TI816X platform.
4. GatePeterson module has been configured with 512 instances.
5. GatePeterson has been plugged into GateMP as GateMP\_RemoteProtect\_CUSTOM1. So applications can create the gate that uses GatePeterson as remote gate by specifying GateMP\_RemoteProtect\_SYSTEM protection level. (For details see GateMP\_RemoteProtect Enum in GateMP.h of IPC package).
6. If the remote gates are not sufficient, applications need to increase the number of instances configured in GatePeterson module both in HLOS side and RTOS side Gatepeterson module.
  1. On HLOS side this can be done by changing the configuration of GatePeterson module in Platform.c file (\$SYSLINK\_ROOT\ti\syslink\family\hlos\kn\omap3530\Platform.c). Refer to the User Guide for more details. For example, see below:

```
var GatePeterson      = xdc.module('ti.sdo.ipc.gates.GatePeterson');
GatePeterson.numInstances = 580;
```

1. No gate is plugged in for GateMP\_RemoteProtect\_CUSTOM2.
2. To change the gate types that are associated with three remote protection levels, application need to rebuild syslink
  1. By doing changes in the file GateMPDefs.h in \$\$SYSLINK\_ROOT/ti/syslink/inc directory of SysLink.
  2. On RTOS side this change has to be done in application's cfg file. For details refer to any syslink examples.

The following code shows how to plugin GatePeterson as RemoteCustom1Proxy in GateMP module to associate GateMP\_RemoteProtect\_CUSTOM1 with GatePeterson.

```
var GatePeterson      = xdc.module('ti.sdo.ipc.gates.GatePeterson');
var GateMP            = xdc.useModule ('ti.sdo.ipc.GateMP');
GateMP.RemoteCustom1Proxy = GatePeterson;
```

- Note:- For Ipc\_attach to pass, the number of instances configured for a gate module (GatePeterson, GateHwSpinlock etc) should be the same on two cores.



**Changes from release 02.00.00.40**

- By default GateMP module's remoteProtection level is GateSpinlock on TI816X and number of these locks are limited to 32. This limits the number of FrameQ instances that can be created if the remote protection level is set as GateMP\_RemoteProtect\_SYSTEM. For FrameQs that works with in processor and between two processors, Apps can use GatePeterson to minimize the consumption of GateSpinLocks.
- The Single core frame sample (\$SYSLINK\_ROOT/ti/syslink/samples/rtos/singlecore/frameq) is updated to demonstrate the plugging in of GatePeterson in GateMP's GateMP.RemoteCustom1Proxy. The following code needs to be added to application's cfg file to plugin gatePeterson.

```
var GatePeterson      = xdc.module('ti.sdo.ipc.gates.GatePeterson');
GatePeterson.numInstances = 512;
var GateMP            = xdc.useModule('ti.sdo.ipc.GateMP');
GateMP.RemoteCustom1Proxy = GatePeterson;
```

- Single core frameq (\$SYSLINK\_ROOT/ti/syslink/samples/rtos/singlecore/frameq) sample is updated to create and run 64 frameQ/FrameQBufMgr instances. Refer this application to pass remote protect paramter to FrameQ\_create/FrameQBufMgr\_create

to ensure that FrameQ/FrameQBufMgr uses GatePeterson for multiprocessor protection.

**Changes from release 02.00.00.39**

- Addition of TilerMemMgr integration into FrameQ
  - For DM740 device, this adds a dependency on Framework Components product.
  - When building RTOS-side, path to Framework Components packages for Tiler and RCM need to be added:
    - export  
XDCPATH="/users/ipc/ipc\_<version>/packages;/toolchains/bios6/bios\_<version>/package
  - Note that \$SYSLINK\_ROOT path has to be added before Framework Components to ensure that this version of SysLink is used instead of the one in Framework Components.
  - The MemMgr package has changed to syslinkMemMgr module:
    - var MemMgr = xdc.loadPackage ("ti.syslink.ipc.rtos.MemMgr"); Changed to:
    - xdc.useModule('ti.syslink.ipc.rtos.syslinkMemMgr.SyslinkMemMgr');
    - xdc.useModule('ti.syslink.ipc.rtos.syslinkMemMgr.shMemMgr.SharedMemoryMgr');
    - xdc.useModule('ti.syslink.ipc.rtos.syslinkMemMgr.nullMemMgr.MemMgrSupportNull');
    - var SharedMemMgr = xdc.loadPackage  
("ti.syslink.ipc.rtos.syslinkMemMgr.shMemMgr");
  - The appropriate syslinkMemMgr package needs to be selected based on the platform.
  - The FrameQ inter-Ducati sample application (frameq\_dm740) has been updated to support both shared memory or tiler based builds based on a Program.global variable (ipcWithHost)
- Move to TI816X PSP based on Linux kernel version 2.6.34
- GateMP sample application has been added
- Documentation generated from cdoc and doxygen has been corrected to ensure that it is generated correctly.



## Known Issues

- Tiler driver is not validated for TI814X as all required dependencies are not available.
- There is a 2 usec delay required only on TI814X between enabling ducati logic and actually accessing the L2RAM. Any access to L2RAM without this delay produces an abort.

Note: There is a work-around for the known issues : SDOCM00078513,SDOCM00078512. Please refer to *Work-around for four core sample app in TI814x* under *Upgrade and Compatibility* section.

## Defects

Defect ID	Headline	Found in Build version
SDOCM00079479	Config.bld is overriding compiler options for M3 targets	
SDOCM00079451	[RingIO] RingIO function test case are failing when RingIO_setWaterMark() API is invoked	DEV_SYSLINK_02.00.00.67
SDOCM00079441	[RINGIO_GPP] RingIO_gpp sample application hangs in Ipc_attach.	DEV_SYSLINK_02.00.00.67
SDOCM00079383	Build system does not cleanly scale to building SysLink for a big endian target	DEV_SYSLINK_02.00.00.67
SDOCM00079368	Memleak issue in 02.00.00.67 of the syslink frameQ	
SDOCM00079280	[Samples] MessageQ sample application incorrectly overshoots message boundary in pattern verification code	DEV_SYSLINK_02.00.00.67
SDOCM00079242	[RingIO] RingIO four core sample application gives Ipc_start failure assertion	DEV_SYSLINK_02.00.00.67
SDOCM00079217	Need to have common memory segment definition for all the rtos cores in syslink samples	DEV_SYSLINK_02.00.00.67
SDOCM00079078	Usability comments on procMgr sample application	DEV_SYSLINK_02.00.00.66
SDOCM00079044	[Notify] NotifyDriverCirc does not handle sendEventPollCount	DEV_SYSLINK_02.00.00.67
SDOCM00079042	[TRACE] SysLink_setup should handle env variable to set appropriate user space traces.	DEV_SYSLINK_02.00.00.67
SDOCM00078974	[Memory Leak] Memory leak is observed in Syslink HLOS side for Syslink.ko and sample applications	DEV_SYSLINK_02.00.00.67
SDOCM00078711	Correct number of gates in SysLink Centaurus code	DEV_SYSLINK_02.00.00.67
SDOCM00078637	[Datasheet] The profiling numbers on A8 should be mentioned in cycles	DEV_SYSLINK_02.00.00.67
SDOCM00078636	[Performance] Lower performance is observed in Netra DDR3 EVM	DEV_SYSLINK_02.00.00.67
SDOCM00078604	SysLink Install Guide documentation incorrectly mentions building ProcMgr knl sample	DEV_SYSLINK_02.00.00.65
SDOCM00078524	Change status codes prints to %d instead of %x	DEV_SYSLINK_02.00.00.66
SDOCM00078521	[FRAMEQ ] interducati frameq samples does not run to completion on ddr3 netra evm	DEV_SYSLINK_02.00.00.66
SDOCM00078513	[SAMPLEAPPS] RingIO four core sample application for TI814X aborts with exception	DEV_SYSLINK_02.00.00.66
SDOCM00078512	[SAMPLEAPPS] FrameQ four core sample for TI814X hangs	DEV_SYSLINK_02.00.00.66
SDOCM00078466	[MsgQ] Performance degradation is observed in MsgQ round trip time.	DEV_SYSLINK_02.00.00.66
SDOCM00078462	[Samples] FrameQ and RingIO samples need to be cleaned up and should follow other sample's approach for maintainability	DEV_SYSLINK_02.00.00.66
SDOCM00078311	ARM load is 95% in kernel space or MessageQ_get returns '-1' based syslink.ko and HDMI.ko insertion sequence	DEV_SYSLINK_02.00.00.65
SDOCM00078181	Tracking review comments in Review 759	DEV_SYSLINK_02.00.00.65
SDOCM00077998	[SAMPLEAPP] GateMp sample app crashes on Tomahawk when executed among 4 or more no of cores	DEV_SYSLINK_02.00.00.56
SDOCM00077848	[GENERIC] Remove references to DM (replace by TI) from code and file names	DEV_SYSLINK_02.00.00.66



SDOCM00077845	[SYSTEM] Infinite loop of Ipc_readConfig should be timed out and return appropriate message to user	DEV_SYSLINK_02.00.00.66
SDOCM00077771	HeapBufMP and HeapMemMP kernel sample application hangs	DEV_SYSLINK_02.00.00.56
SDOCM00077399	[Generic] User-side SysLink_setup must read trace information from environment and setup trace internally	DEV_SYSLINK_02.00.00.66
SDOCM00077367	[Ipc] Ipc_readConfig does not free shared memory for the entry after returning it to the user	DEV_SYSLINK_02.00.00.66
SDOCM00077190	Build step for building hlos user side procMgr sample application is wrong	DEV_SYSLINK_02.00.00.65
SDOCM00077185	[SYSTEM] CLASS4 trace prints needs to be added to critical functions	DEV_SYSLINK_02.00.00.65
SDOCM00077183	[RINGIOGPP] RingioGpp kernel side sample application hangs on TI814x	
SDOCM00077149	[Notify] Notify lower round trip performance is observed between CortexA8 and Video-M3 on Centaurus	DEV_SYSLINK_02.00.00.65
SDOCM00077128	ProcLoad should return error instead Load call back on Memory map issues	DEV_SYSLINK_02.00.00.65
SDOCM00077038	[RingIO] In RingIO sample applications attrSharedAddrSize is configured twice which is not necessary.	DEV_SYSLINK_02.00.00.65
SDOCM00077037	[FRAMEQ] SegFault seen while running FrameQ test suite on OMAP3530	DEV_SYSLINK_02.00.00.65
SDOCM00077035	[FRAMEQ] Iommu fault seen on OMAP3530 while running FrameQ test suite	DEV_SYSLINK_02.00.00.65
SDOCM00077031	[SysLink] recursive lock warnings are observed with 2.6.36 kernel built with omap2plus_defconfig	DEV_SYSLINK_02.00.00.65
SDOCM00077030	[FrameQ] API is needed to set the module gate that can be used by instances if useDefaultGate is set to TRUE in module	DEV_SYSLINK_02.00.00.65
SDOCM00077022	SharedRegion kernel sample module fails	DEV_SYSLINK_02.00.00.56
SDOCM00076881	Calling Mod_destroy before calling Mod_setup causes incorrect reference count and subsequent destroy call to fail	DEV_SYSLINK_02.00.00.65
SDOCM00076878	[RingIO] ringIO sample shows assertions when run alternatively with heapmemmp sample	
SDOCM00076871	[RingIO] Application crash is observed when RingIO instance is tried to open twice with same name in HLOS side.	DEV_SYSLINK_02.00.00.65
SDOCM00076836	[SampleApp] FrameQ and RingIO sample applications should support data transfer operation from RTOS to HLOS also.	DEV_SYSLINK_02.00.00.65
SDOCM00076748	[Performance] Lower round trip performance is observed in MsgQ between CortexA8 and Video-M3 on Centaurus	DEV_SYSLINK_02.00.00.65
SDOCM00076554	[RingIO] RingIO_setAttribute API fails when invoked after buffer is completely acquired and released completely	DEV_SYSLINK_02.00.00.65
SDOCM00076495	RingIO single core app fails in RingIO_Open on dsp of Centaurus	DEV_SYSLINK_02.00.00.65
SDOCM00076425	[RINGIO] RingIO_close() returns success even if handle passed is NULL	DEV_SYSLINK_02.00.00.65
SDOCM00076395	RingIO code review defects	DEV_SYSLINK_02.00.00.65
SDOCM00076306	[RingIO] RingIO code review comment	DEV_SYSLINK_02.00.00.65
SDOCM00076039	[Generic] MemoryOS_map and unmap do not check for size and range when reference counting existing mapped regions	DEV_SYSLINK_02.00.00.65
SDOCM00075792	[RingIO] API profiling numbers increases in every run for RingIO_create and RingIO_open APIs.	DEV_SYSLINK_02.00.00.64
SDOCM00075791	[RingIO] SysLink RingIO API profiling numbers are higher than DspLink RingIO API profiling numbers on OMAP3530 platform	DEV_SYSLINK_02.00.00.64
SDOCM00075737	ProcMgr ELF loader does not support argc/argv loading	DEV_SYSLINK_02.00.00.53
SDOCM00074925	[SAMPLE APP] Assertions seen in RingIO_gpp when run in a script along with other samples.	DEV_SYSLINK_02.00.00.53



SDOCM00074803	[SysLink] Build warnings are observed during COFF format samples build for DSP of ti816x	DEV_SYSLINK_02.00.00.53
SDOCM00074765	[ProcMgr] ProcMgr_MAX_MEMORY_REGIONS needs to be increased from 32 and print additional info when entries are used up	DEV_SYSLINK_02.00.00.53
SDOCM00074355	Detect if memory used by other cores extends into the memory allotted to Linux	DEV_SYSLINK_02.00.00.53
SDOCM00074137	[ProcMgr] Should use Linux clk framework for OMAP3530 and OMAPL1xx code also	DEV_SYSLINK_02.00.00.53
SDOCM00073900	[Generic] Test execution takes long time in DSP processor of NETRA EVM (Task Sleep take long time)	DEV_SYSLINK_02.00.00.52
SDOCM00073685	[Generic] Ipc.xdt config info needs to be moved into SysLink product	DEV_SYSLINK_02.00.00.51
SDOCM00073227	Not verifying the create time paramters with Max possible values.	
SDOCM00072929	Getting asserts but the APIs return success	
SDOCM00072921	[RingIO] Sometimes "RINGIO_NOTIFICATION_HDWRFIFO_ALWAYS" notification doesn't occur in reader mode on DSP	DEV_02.00.00.44
SDOCM00072864	[FrameQ] FrameQ_putv API crashes in HLOS side	DEV_SYSLINK_02.00.00.50
SDOCM00072559	Syslink dis- allow the linux to report error message to user space application which makes debugging very difficult	
SDOCM00071882	[FrameQ] FrameQ notification are not received on ARM side when sent from M3VIDEO on Netra EVM	DEV_02.00.00.44
SDOCM00071384	[OS Port] Osal_printf() call gives segmentation fault if string length is more than 4060 characters	DEV_02.00.00.44
SDOCM00070872	Intra A8 FQ on needs the slave core to be loaded & Executed	
SDOCM00070862	User space application crash causes linux to reboot	
SDOCM00070712	ProcMgr_load asserts inappropriate error message (invalid ELF file) for missing file	DEV_02.00.00.44
SDOCM00070324	LINK - Require a mechanism to get the number of LINK interrupts	
SDOCM00069082	[RingIO] RingIO Notification (RingIO_NOTIFICATION_ALWAYS) doesn't occur on ARM (in writer and reader modes)	DEV_02.00.00.39
SDOCM00068969	[GateMP] Incorrectly implemented or partially implemented helper functions on user-side	DEV_02.00.00.39
SDOCM00068960	[ListMP] ListMP_getGate on user-side is not implemented	DEV_02.00.00.39
SDOCM00068875	[RingIO] RingIO create [RingIO_create() API] crashes if shared region is not set [SharedRegion_setEntry() API] on ARM	DEV_02.00.00.39
SDOCM00068597	[GateMP] Default GateMP_LocalProtect value is set to GateMP_LocalProtect_INTERRUPT enum instead of GateMP_LocalProtect_THREAD	DEV_02.00.00.38
SDOCM00068411	[RINGIO] Code review comments	DEV_02.00.00.38
SDOCM00068237	[RingIO] Assertion failure is seen on DSP when RingIO_setvAttribute() API is called with sendNotification flag set to TRUE	DEV_02.00.00.27
SDOCM00068235	[RingIO] "RINGIO_S_PENDINGATTRIBUTE" Macro define must be a negative value	DEV_02.00.00.27
SDOCM00068226	[Generic] MISRA C compliance reports to be available for the product	DEV_02.00.00.37
SDOCM00068225	[RingIO] Assertions are observed in RingIOTransportShm_delete () API call on Linux (ARM) in stability test	DEV_02.00.00.27
SDOCM00068224	[RingIO] RingIO_getvAttribute() API call fails if called in the same thread as that of RingIO writer	DEV_02.00.00.27
SDOCM00068210	[FrameQ]: FrameQBufMgr_create() API should not allow to create instances more than FrameQBufMgr_MAXINSTANCES	DEV_02.00.00.37
SDOCM00068170	[MessageQ] MessageQ_put() results in kernel crash when the message queue has already been deleted	DEV_02.00.00.37



SDOCM00068120	[RingIO] Programmers guide and design document doesn't give enough information on RingIO APIs and their parameters	DEV_02.00.00.27
SDOCM00068061	[RingIO] RingIO Notification (RingIO_NOTIFICATION_ONCE) doesn't occur on DSP when opened as reader	DEV_02.00.00.27
SDOCM00068034	[FrameQ] FrameQ_registerNotifier API returns assertion when called multiple times	DEV_02.00.00.27
SDOCM00067988	[Generic] Sample application functionality description must be documented	DEV_02.00.00.27
SDOCM00067981	[RingIO] If "RingIOApp_MULTIPROCESS" is defined in the RingIO sample app, hlos user build fails	DEV_02.00.00.27
SDOCM00067965	[FrameQ] FrameQ should return error when offset+valid size is more than frame buffer size	DEV_02.00.00.27
SDOCM00067961	[Build] BUILD_OPTIMIZE should not compile out run time resource allocation error checks	DEV_02.00.00.27
SDOCM00067846	[Notify] Notify Reg/ Un-reg event APIs return success & Kernel (crash) prints are seen if Line Id is greater than 0	DEV_02.00.00.27
SDOCM00067798	[MessageQ] Requirement SR196 not implemented	
SDOCM00067780	[MessageQ] Requirement SR188 not implemented	
SDOCM00067778	[Notify] The Notify design document and requirements have not been updated as per the new design	
SDOCM00067768	[MessageQ] High level design document needs to be updated	
SDOCM00067766	[MessageQ] MessageQ_alloc() should check minimum memory required for a Message allocation.	
SDOCM00067760	[MessageQ ] MessageQ - code review comments.	DEV_02.00.00.27
SDOCM00064032	SysLink driver needs reference counting when mapping memory blocks into kernel space	
SDOCM00064026	ProcMgr_unmap() should return error if called after ProcMgr_detach()	
SDOCM00063898	ProcMgr_unmap() should take address not AddrInfo structure	
SDOCM00063680	Assert raised in SharedRegion_add when called by multiple programs	
SDOCM00061571	In HeapMultiBuf, combining buckets of same size does not take alignment constraints into account	DEV_2.00.00.01
SDOCM00061570	Improper kbuild makefile structure	

## Versioning

This is the SysLink\_02.00.00.68\_beta1 release

## Technical Support and Product Updates

Procedure to report problems/issues for TI products is available at the TI support page:

- External forums <sup>[14]</sup> Or:
- <http://support.ti.com> <sup>[15]</sup>

For TI81xx support:

- [http://e2e.ti.com/support/dsp/davinci\\_digital\\_media\\_processors/int-dm81x/default.aspx](http://e2e.ti.com/support/dsp/davinci_digital_media_processors/int-dm81x/default.aspx) <sup>[16]</sup>



## References

- [1] [http://www.sanb.design.ti.com/tisb\\_releases/BIOS/6\\_31\\_04\\_27/](http://www.sanb.design.ti.com/tisb_releases/BIOS/6_31_04_27/)
  - [2] [http://www.sanb.design.ti.com/tisb\\_releases/XDCtools/3\\_20\\_08\\_88/](http://www.sanb.design.ti.com/tisb_releases/XDCtools/3_20_08_88/)
  - [3] [http://www.sanb.design.ti.com/tisb\\_releases/IPC/1\\_22\\_04\\_25/](http://www.sanb.design.ti.com/tisb_releases/IPC/1_22_04_25/)
  - [4] <http://syntaxerror.dal.design.ti.com/release/release.cgi>
  - [5] [http://tiexpressdsp.com/wiki/index.php?title=Category:Code\\_Composer\\_Studio\\_v4](http://tiexpressdsp.com/wiki/index.php?title=Category:Code_Composer_Studio_v4)
  - [6] <http://www.codesourcery.com/sgpp/lite/arm/portal/release858>
  - [7] [ftp://ftp.india.ti.com/PSP/Releases/ODC/OMAP35XX\\_GIT\\_PSP/AM35x-OMAP35x-PSP-SDK-03.00.00.03.tgz](ftp://ftp.india.ti.com/PSP/Releases/ODC/OMAP35XX_GIT_PSP/AM35x-OMAP35x-PSP-SDK-03.00.00.03.tgz)
  - [8] [http://www.sanb.design.ti.com/tisb\\_releases/BIOS/6\\_31\\_04\\_27/](http://www.sanb.design.ti.com/tisb_releases/BIOS/6_31_04_27/)
  - [9] [ftp://ftp.india.ti.com/PSP/Releases/ODC/Netra\\_GIT\\_PSP/TI816X-LINUX-PSP-04.00.00.10.tgz](ftp://ftp.india.ti.com/PSP/Releases/ODC/Netra_GIT_PSP/TI816X-LINUX-PSP-04.00.00.10.tgz)
  - [10] [ftp://ftp.india.ti.com/PSP/Releases/ODC/Centaurus\\_PSP/TI814X-LINUX-PSP-04.01.00.03.tgz](ftp://ftp.india.ti.com/PSP/Releases/ODC/Centaurus_PSP/TI814X-LINUX-PSP-04.01.00.03.tgz)
  - [11] [http://www.sanb.design.ti.com/tisb\\_releases/Framework\\_Components/3\\_21\\_00\\_15/](http://www.sanb.design.ti.com/tisb_releases/Framework_Components/3_21_00_15/)
  - [12] [http://www.codesourcery.com/gnu\\_toolchains/arm/portal/package1786?@template=release](http://www.codesourcery.com/gnu_toolchains/arm/portal/package1786?@template=release)
  - [13] <ftp://ftp.india.ti.com/PSP/Releases/ODC/DaVinci-PSP-SDK/DaVinci-PSP-SDK-03.20.00.11.tgz>
  - [14] <http://community.ti.com>
  - [15] <http://support.ti.com>
  - [16] [http://e2e.ti.com/support/dsp/davinci\\_digital\\_media\\_processors/int-dm81x/default.aspx](http://e2e.ti.com/support/dsp/davinci_digital_media_processors/int-dm81x/default.aspx)
-



# Article Sources and Contributors

**SysLink 02.00.00.68 beta1 ReleaseNotes** *Source:* <http://ap-fpdp-swapps.dal.design.ti.com/index.php?oldid=91101> *Contributors:* ChrisRing, DeepaliUppal, Goutamkumar, MugdhaKamoolkar, NagabhushanReddy, Ravindranath Andela, YogeshMarathe